Data Splash!
An Educational Game about Machine Learning

Andrew Gray
445348

Submitted to Swansea University in fulfilment
of the requirements for the Degree of Master of Science

Swansea University
Prifysgol Abertawe

Department of Computer Science
Swansea University

September 20th, 2020

Declaration

This work has not been previously accepted in substance for any degree and is not being concurrently submitted in candidature for any degree.

Signed ..... (candidate)

Date .....

Statement 1

This thesis is the result of my own investigations, except where otherwise stated. Other sources are acknowledged by footnotes giving explicit references. A bibliography is appended.

Signed ..... (candidate)

Date .....

Statement 2

I hereby give my consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signed ..... (candidate)

Date .....

I would like to dedicate this work to the Hypnotoad.
All glory to the Hypnotoad.

Abstract

In your abstract you should aim to summarize the core contributions of your work in the context of the problem domain. Start by outlining the domain and the problems posed within it. Discuss how the methods you focus on approach the relevant problems. You should end your abstract by concretely stating the tangible outputs and deliverables you have created in order to complete your work on this document, and whether those outputs represent and improvement or alternative approach to existing methods.

Your abstract should be a couple or so paragraphs long, and roughly approximate the order and flow you then use for structuring the main document. If a viewer has read your abstract then they should already understand at a high level what it is you have created and delivered, and whether it is better than or comparable to existing methods. If your project is driven by a research hypothesis then the reader should know what that is at a high level from this section. Reading on, little should surprise the viewer.

For paper submission of your thesis you should physically sign your name on each of the above declaration statements and date them in black ink. For digital submissions you should sign and date them digitally using a touch or stylus input if available. There are pieces of software that allow you to write directly on PDF documents, or alternatively you can bring a signature into your document as a figure with a transparent or white background. If you do not have a stylus input / tablet like device you should ask your supervisor, as many in the department do their grading / work on digital tablets.

Acknowledgements

This is an opportunity to acknowledge and thank those who have supported you throughout your studies. Friends and colleagues who you have studied alongside, your families, and your mentors within the department are the usual suspects.

Contents

List of Tables **vii**

List of Figures **viii**

1 Introduction **1**

1.1 Motivations 1

1.2 Overview 2

1.3 Contributions 2

2 Background & Literature Review **3**

3 Methodology **5**

3.1 Overview of Application 5

3.2 Overview of Specific Game Components 6

3.3 Evaluation of Application 6

4 Implementation **9**

4.1 Tools 9

4.2 Referencing items within this document 12

4.3 Equations 12

4.4 Figures 14

4.5 Code Listings 18

4.6 Tables 19

5 Conclusions and Future Work **21**

5.1 Contributions 21

5.2 Future Work 21

Bibliography **23**

Appendices **24**

A Implementation of a Relevant Algorithm **25**

B Supplementary Data **27**

List of Tables

4.1 A demonstration of a table typeset in LaTeX. 19

List of Figures

4.1 A comparison between Java, Python and C++ to print an output to the console. [1] . 10
4.2 A screenshot of TeXnique, a game about typesetting equations. 14
4.3 An image of many glass dragons being used to demonstrate typesetting a figure. . . 16
4.4 A demonstration of a 2x1 sub-figure layout. 17
4.5 A demonstration of a 2x2 sub-figure layout. 17

Chapter 1

Introduction

This document is intended both as a thesis template and a written tutorial on typesetting a professional looking academic document. The style of the template is designed to mimic an equivalent LaTeX document template that is commonly used for within the Computer Vision and Visual Analytics group here at Swansea. This LaTeX template is itself based on a LaTeX template named Custard.

1.1 Motivations

Large documents can become cumbersome to work with and format consistently. Sensibly chosen aesthetic cues are important to help imply structure and can greatly aid the reader in understanding your work. The accompanying LaTeX template uses abstraction to hide the formatting from the author during content preparation, allowing for consistent styling to be applied automatically during document compilation. In this Google Docs theme it is the responsibility of the author to manually adhere to the styling laid out in this template.

1.1.1 Objective

In this document we present a tutorial on thesis creation and typesetting, and discuss topics such as literature surveying and proper citation.

1.2 Overview

The remainder of chapter 1 outlines the document structure and the key contributions of this work is organized as follows. Chapter **??** reviews techniques for finding and properly citing external resources from the academic literature and online. In chapter **??** we show examples of how to typeset different types of content, such as internal references, figures, code listings, and tables. And lastly in chapter 5 we summarize the main contributions and key points to take away from this template.

1.3 Contributions

The main contributions of this work can be seen as follows:

- **A LaTeX thesis template**

  Modify this document by adding additional TeX files for your top level content chapters.

- **A typesetting guide of useful primitive elements**

  Use the building blocks within this template to typeset each part of your document. Aim to use simple and reusable elements to keep your LaTeX code neat and to make your document consistently styled throughout.

- **A review of how to find and cite external resources**

  We review techniques and resources for finding and properly citing resources from the prior academic literature and from online resources.

Chapter 2

Background & Literature Review

Chapter 3

Methodology

The university has subscriptions to a vast number of major academic journals spanning a wide range of subject areas. By accessing the internet from a university network connection (Eduroam or Ethernet), the paywalls of many journals will simply vanish without any need for login credentials.

3.1 Overview of Application

When you are working from outside of the university then connecting to an on campus machine via remote desktop (RemoteDesktopProtocol, TeamViewer, ect) or via port forwarding (ssh, ssh tunnel, ect) can allow you to access papers that would otherwise be behind a paywall.

If you do not have individual access to a machine that is exposed for ssh on the university network you can always use the computers in Linux Lab CF204[1] for the purpose of setting up an ssh port tunnel to proxy your internet through. These machines have fixed IPv4 addresses and respond to ssh using your student account credentials. While in use your internet will be routed[2] to the university and then out to the internet, granting you transparent access to journals without a paywall.

[1] One caveat of using computer lab machines for remote tunnelling is that a environmentally conscious student who has worked late in the computer lab might choose to switch off the machine you were using...

[2] Painfully slowly.

3.1.1 Design

3.2 Overview of Specific Game Components

The internet is big [2]. Knowing how to phrase a question to a search engine is therefore an invaluable skill. If the request is simple enough, even a poorly structured query will likely return usable results. For more difficult to find resources you can leverage the language of the search engine to gather relevant papers and resources for your research more efficiently.

`https://www.gwern.net/Search`

"Internet Search Tips" [3] provides an excellent review of methods and tips for scouring the internet for hard to find resources. You will also be less likely to get caught behind journal paywalls when working remotely without a tunnel as your queries can be made to look for raw pdfs that are often released by the authors directly.

3.2.1 Game Arena

3.2.2 Free Play

3.2.3 Learning Zone

3.2.4 Awards Zone

3.3 Evaluation of Application

BibTeX is a language for specifying resource citations. Every time you access and read an academic paper, take code from an online repository, or source the media such as images from existing works you should create a BibTeX entry in a file that you keep throughout your research. Software such as Mendeley [4] can help automate the process of building your BibTeX library of citations.

```
1 @INPROCEEDINGS{kaj86,
2     author    = {Kajiya, James T.},
3     title     = {The Rendering Equation},
4     booktitle = {Proceedings of the 13th Annual Conference on Computer Graphics
          and Interactive Techniques},
5     year      = {1986},
6     series    = {SIGGRAPH '86},
7     pages     = {143--150},
8     address   = {New York, NY, USA},
```

```
    publisher = {ACM},
    isbn      = {0-89791-196-2},
    numpages  = {8},
    acmid     = {15902}
}
```

Listing 3.1: An example BibTeX entry for an academic paper published in conference proceedings [5].

The BibTeX code listing above (listing 3.1) shows an example of how to cite an academic paper, in this case one of the central papers in Computer Graphics research. The key **kaj86** is an arbitrary name chosen as a meaningful identifier for the resource. In the document text we can call on this resource as an inline citation using the LaTeX command `\cite{kaj86}` which produces [5] at the location it is called. As long as a citation has been used at least once somewhere within the document then a formatted full citation will be created in the bibliography at the end of the document with the same citation number that is shown inline.

It is considerably easier to be disciplined in methodically taking note of the resources you access and make use of as you access them, than it is to try and hunt them all down again at the time you need to write about them in your document. Invest time in being organized and consistent up front and it will be easier when you come to write up.

3.3.1 User Study

Chapter 4

Implementation

4.1 Tools

4.1.1 Programming Languages

For the implementation of our application, three primary programming languages deemed to be best suited for development. Apple's Swift programming language [6] got considered early on, due to the author's familiarisation with the programming language. The programming language gets used for creating applications for Apple's mobile and desktop operating systems, and with 1.5 billion [7] iOS devices in circulation, that was a lot of potential users. Additionally, Apple's iOS devices are prevalent within most educational settings, with Apple's iPad being one of the primary go-to devices. However, due to the language not supporting key frameworks required, or providing similar alternatives, the decision to not use this language got made.

We then got presented with three main options to use, Python, R and HTML, CSS and JavaScript.

Python is a very popular programming language [8, 9], it is fast, easy-to-use, and easy-to-deploy programming language that gets widely used to develop scalable applications. Examples include YouTube, Instagram, Pinterest and SurveyMonkey [10]. The Python Software Foundation state that Python is a high-level object-orientated interpreted language with dynamic semantics. Due to the language being a high-level, it has many built-in data structures. These features, along with the dynamic typing and dynamic binding together make Python attractive to development teams working in a Rapid Application Development (RAD). As Python is an extracted level above the C language [11], Python can get used as the glue that connects existing components, as well as being able to be used as a scripting language [12]. Python gets

considered to be easy to learn the language due to its high readability and is recommended by many exam boards as the language to use for teaching Computer Science at GCSE and A-Level level [**?**]. Python's simple and easy to learn syntax emphasises on readability, which, as a result, reduces the cost of program maintenance [12].

Python gets compared to a lot of other languages. However, due to the requirements and expectations of the application, we will compare it to other similar style applications that can potentially do a similar job. These being Java, JavaScript and C++. In general, the choice of the programming language to use is many other real-world constraints, for example, financial cost, availability, training and even personal preferences and attachments. However, we will focus on language issues for the comparisons.

In comparison to Java, Python programs will typically take 3-5 times quicker (See fig:4.1) to develop but will have a slower run time. The time difference gets attributed to Python's built-in data types and its dynamic typing [13].

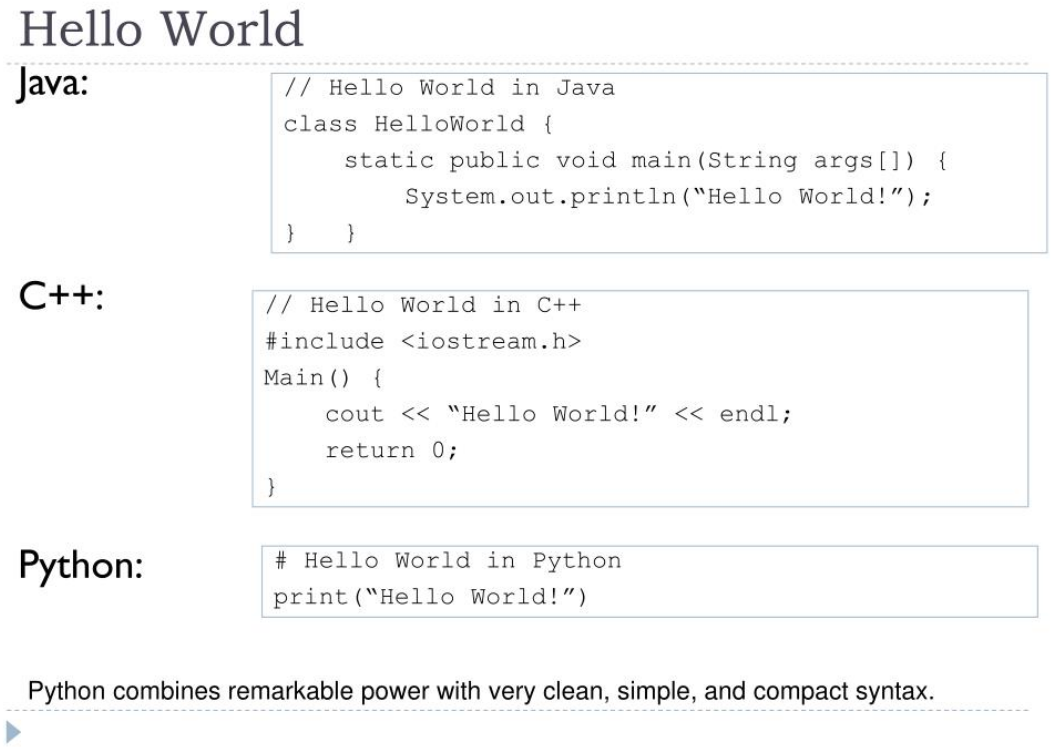


Figure 4.1: A comparison between Java, Python and C++ to print an output to the console. [1]

This document is intended as both a LaTeX thesis template and as a tutorial on structuring and typesetting your thesis in the LaTeX programming language.

The following are some powerful online resources for learning about LaTeX:

- **Overleaf Documentation for LaTeX**

  Overleaf [14] is an online browser-based LaTeX IDE which stores your document in the cloud and provides live recompilation as you type. The documentation on Overleaf's website has a good knowledge base of examples for how to typeset things cleanly and simply in LaTeX code.

  See: `https://www.overleaf.com/learn`

- **TeX StackExchange, the StackOverflow site dedicated to TeX questions**

  TeX StackExchange [15] is sub-community of the StackOverflow network dedicated to questions about the TeX family of typesetting tools including LaTeX, BibTeX and others. A vast majority of the time it is unlikely that the question or issue you are facing is one that has not been encountered before, and this site more than likely to be able to point you in the correct direction.

  See: `https://tex.stackexchange.com`

4.2 Referencing items within this document

In section **??** we saw examples of how to typeset citations for resources we had stored in an external BibTeX file. However, often we would like to accurately refer to the location of a resource or region of text stored somewhere else within this document[1]. To do this we need to annotate our LaTeX code with `\label{key}` statements which will take on the numeric (or otherwise formatted) identifier for the current chapter, section, figure, table, equation, ect where they are directly defined. To insert an inline reference to the label you can use the `\ref{key}` command which works similarly to the `\cite{key}` used for external references. In the event we chose to reorder or add additional content to the document, which would change the section numbering, the document will still compile to a pdf with the correct references inserted for each `\ref{key}` command.

4.3 Equations

Typesetting equations is one of the things that LaTeX does best. It has packages for different fonts and symbols for many different mathematical notations. However, to person learning how to typeset in LaTeX for the first time it can be a daunting and unwieldy user experience. Almost all LaTeX packages have documentation available in pdf format online, and documentation for packages specifically relating to fonts and symbols usually have tables enumerating the names and codes for all of the fonts symbols, organized by intended usage.

4.3.1 Inline equations

Small equations like $x = 0$ can be written directly within the text by using LaTeX's maths mode shorthand controlled by dollar signs `$ math mode $`. As long as it is not becoming cumbersome to the reader, equations such as $\mathbb{P}(A \cap B) = \mathbb{P}(B \cap A)$ are quite neatly displayed in this fashion.

[1] Like at the beginning of the last sentence when we referred to section **??**.

4.3.2 Block equations

For long equations it is best to provide a break in the main text of the document and format the equation using a `\begin{equation}...\end{equation}` environment.

$$|a| = \left\| \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} \right\| = \sqrt{a_0^2 + a_1^2 + \ldots + a_n^2} \tag{4.1}$$

Equation 4.1 demonstrates formatting a larger equation and uses an `\begin{array}...\end{array}` environment to structure a column vector of sub-equations. Block equations should be located at a relevant point directly as they are being referred to in the text. When referred to from other locations in the document you should use the `\ref{key}` command to insert the correct equation number.

4.3.2.1 Aligning multi-line block equations

When equations become even larger they may need cross over multiple new lines. When this happens it is desirable to align relevant parts of the equation on each line to one another for aesthetic reasons and to help imply structure to the reader.

$$\begin{aligned} \mathscr{L}_o(x,\omega_o,\lambda,t) &= \mathscr{L}_e(x,\omega_o,\lambda,t) \\ &\quad + \int_\Omega f(x,\omega_i,\omega_o,\lambda,t)\,\mathscr{L}_i(x,\omega_i,\lambda,t)\,(\omega_i \bullet n)\,d\omega_i \\ &\quad \text{where} \quad \mathscr{L}_i(x,\omega_i,\lambda,t) = \mathscr{L}_o\left(x',-\omega_i,\lambda,t\right) \end{aligned} \tag{4.2}$$

Equation 4.2, known as Kajiya's Rendering Equation [5] demonstrates the use of the `\begin{split}...\end{split}` environment which uses a single un-escaped & symbol placed on each line of the equations LaTeX code to indicate where each line should be co-aligned. In this example the &'s were placed on the =, +, and w (in where) characters.

4.3.3 A masochistic approach to learning to typeset mathematics in LaTeX

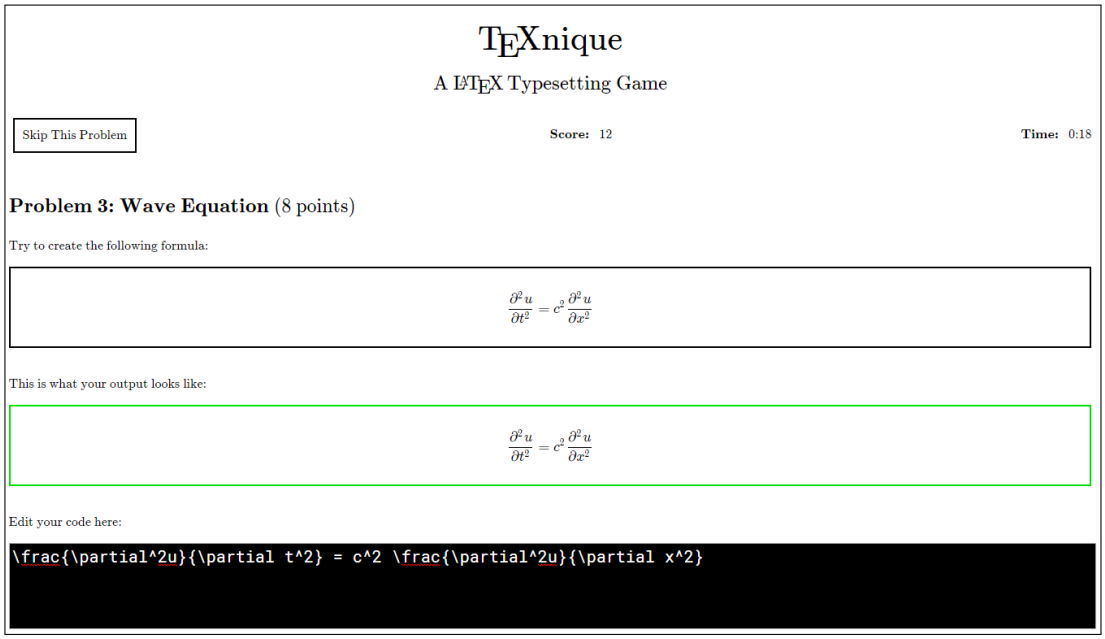


Figure 4.2: TeXnique, a game about typesetting equations [16]. (Top) The game presents you with a rendered equation, (Bottom) the task is to enter LaTeX code that produces the same rendered equation. The green border on the lower rendering indicates it is a valid solution.

TeXnique [16] is web-browser based game for practising how to typeset equations in LaTeX. The game will present you with a rendered equation and your task is to type LaTeX code into the box below it such that your code produces the same (or closely matching / pixel equivalent) rendered equation. Figure 4.2 shows the game during play, the bottom rendered equation is bordered in green to indicate it is a valid match with the target.

`https://texnique.xyz`

This is one of the more painful parts of typesetting a document, so it really takes a special kind of sadism to come up with such a game. Least to say, graduate students and researchers can be an odd bunch, and when we found this it was surprisingly addictive to compete over.

4.4 Figures

In this template figures are numbered starting with the current chapter number followed by a figure number that resets to 1 each new chapter. As you can see below, the first figure is

labelled Figure 4.3 because we are in Chapter **??**.

Figures in LaTeX are defined using a `\begin{figure}...\end{figure}` environment and often immediately begin rendering in centre aligned mode by calling `\centering`. Listing 4.1 below shows the LaTeX code used to typeset figure 4.3. Figures 4.4 and 4.5 are defined similarly and make additional use of the `\subfloat` command to position multiple images within a single figure environment, each with their own automatically incremented labels and individual captions.

```
\begin{figure}[H]
  % [H] means put the figure HERE, directly when you input this code.
  \centering

  % We set the width of the figure based on the width of one line
  % of text on the page. The value can be tuned to any value in
  % [0.0, 1.0] to scale the image while maintaining its aspect ratio.
  \includegraphics[width=1.0\linewidth]{./graphics/dragon.png}

  % Caption is defined with a short and long version. The short
  % version is shown in the List of Figures section, and the long
  % version is used directly with the figure.
  \caption[Short caption.]{Long caption and citation \cite{whittle15_dragons}.}

  % For figures, \label should be defined after the caption to ensure
  % proper figure numbering.
  \label{fig:dragon}
\end{figure}
```

Listing 4.1: An example LaTeX excerpt demonstrating how to typeset figure 4.3 with a simple caption.

4.4.1 Consistent presentation throughout the document

Figures work best in a document when you use a consistent style for formatting and captioning them and make sure that figures always actively support the content of the main text.

4.4.2 Justified use of space in the document

All figures must be referred to directly in the main text of the document and discussed with meaningful and in depth critical analysis. If you don't need to use the figure to leverage and support your discussion then it is just taking up space and padding out the document. For example, you can use a command like `\ref{fig:dragon}` to automatically get the figure number for Figure 4.3.

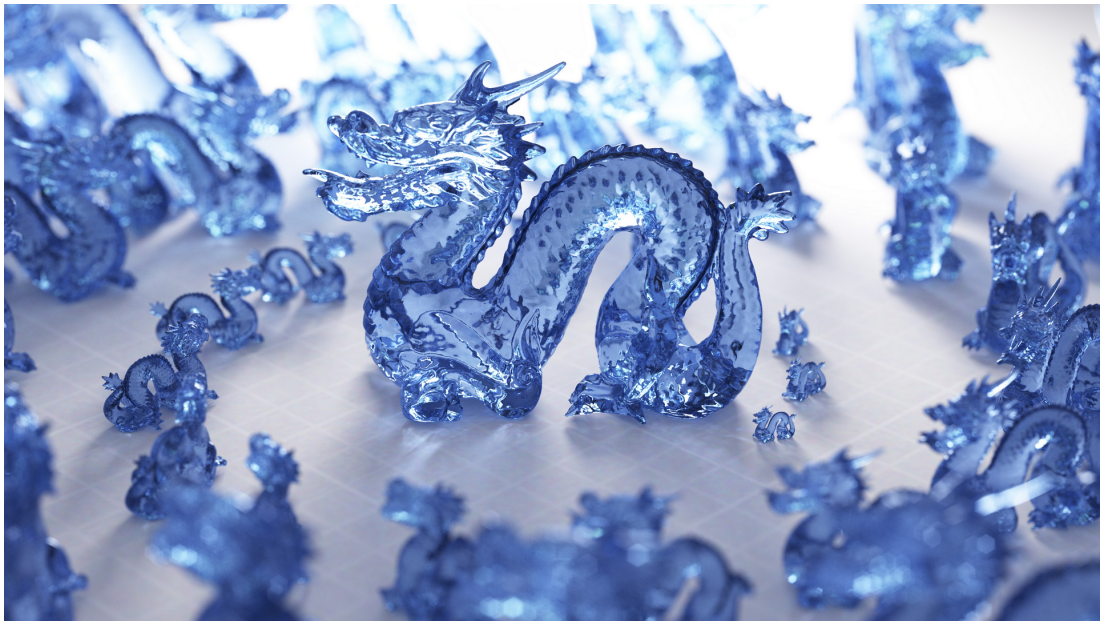


Figure 4.3: A good caption should be sufficient enough to put the figure in context even if the reader has randomly flicked to the current page and looked only at the figure in isolation. All figures should also be referred to directly within the main text of your document. You can use the LaTeX **\ref{key}** command to insert the correct figure number when you refer to it in the main text. By the very logic of this caption, this is a very poor caption because we still don't know why on earth is there an picture of glass dragons here. Image of glass dragons rendered using Path Tracing [17].

4.4.3 Placement that supports and enhances the flow of the document

All figures shown in your document should be displayed in relevant locations, ideally just after that have been alluded to in the main text. Although there are many times where it is best to force a figure to the top or bottom of a nearby page.

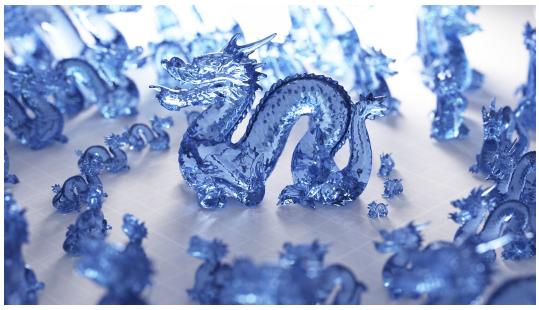
4.4.4 Avoid directly importing other peoples images

You should avoid using other peoples figures whenever possible, and instead create your own figures for visualizing the specific methods and data you are working with in a way directly relevant to your project.

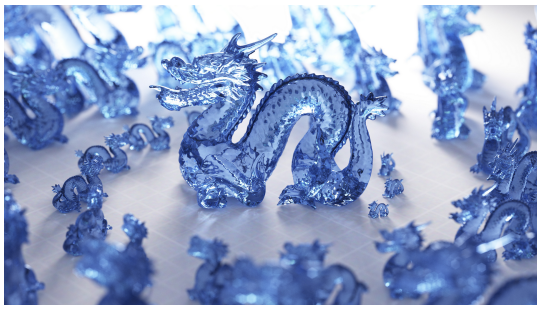
4.4.5 Format sub-figures in LaTeX, not in the image itself

Construct sub-figures from multiple image files in LaTeX not in the image file itself. This allows you to tweak the positioning and layout without having to modify the images. It also

allows for automatic formatting and numbering of captions and sub-captions. Figures 4.4 and 4.5 show examples of side-by-side and quad layouts respectively.

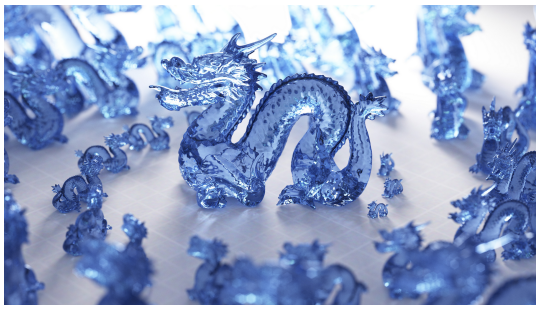


A. Left image sub-caption.

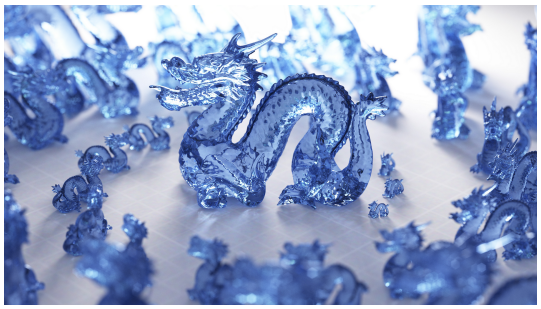


B. Right image sub-caption.

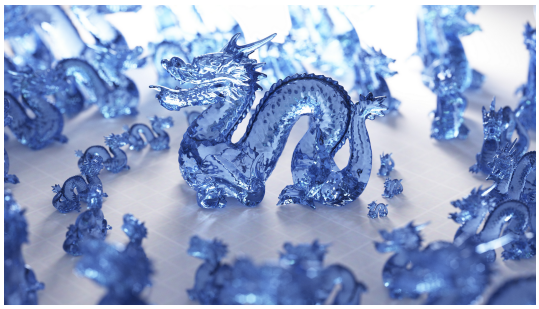
Figure 4.4: Construct sub-figures from multiple image files in LaTeX not in the image file itself. This allows you to tweak the positioning and layout without having to modify the images. It also allows for automatic formatting and numbering of captions and sub-captions. Image of glass dragons rendered using Path Tracing [17].



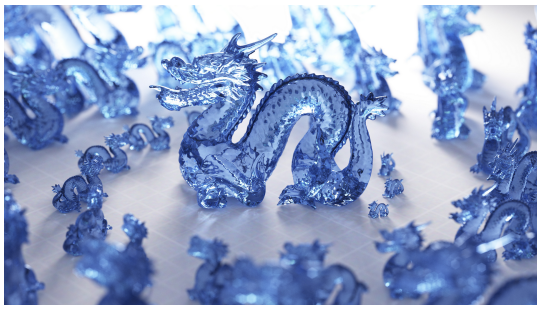
A. Top-Left image sub-caption.



B. Top-Right image sub-caption.



C. Bottom-Left image sub-caption.



D. Bottom-Right image sub-caption.

Figure 4.5: A demonstration of a 2x2 sub-figure layout. Between A-B and C-D we use tilde symbols and between B-C we use a new line. Image of glass dragons rendered using Path Tracing [17].

4.4.6 Robust captions that can stand in isolation

Figures need to be captioned such that they can be viewed in isolation and still be meaningful to the viewer. There will likely be some duplication of information that is written in the main text, but this is intended.

4.4.7 Proper attribution and citation of images

If an image does not belong to you it **must** be cited directly in the figure caption. **It is not correct to put a URL in the figure caption directly.** A URL in isolation is not an accurate or reliable way of directing a future reader to the exact content you are referencing. Instead make a new entry in your `citations.bib` file and then reference that citation in the caption using the `\cite{key}` command. Figures 4.3, 4.4, and 4.5 each include a statement in the caption stating "Image of glass dragons rendered using Path Tracing [17].". When adding the BibTeX entry, try to find the proper information about the original author and source document to strengthen the citation in case the URL changes.

4.5 Code Listings

Code listings should be formatted in the same style as figures and inline equations. It is important to use a monospace font so that characters line up vertically. Syntax highlighting is also extremely important for effectively displaying complicated code segments. To format inline code listings you can use the `\lstinline|the_code|` command[2].

```
#include <stdio.h>

int main(int argc, char *argv[]) {
  printf("Hello world.\n");
  return 0;
}
```

Listing 4.2: An implementation of an important algorithm from our work.

In LaTeX the "Listings" package can be used to properly format code and provide basic syntax highlighting, line numbering, and captioning of embedded code excerpts. Listing 4.3 shows examples of how to properly format code using the listings package.

[2] So meta.

```
% The lstinline command can be used to insert monospace formatted code directly
% inline within the documents main text. You can optionally specify a programming
% language to enable syntax highlighting.
\lstinline|the_code|
\lstinline[language={the_language}]|the_code|

% The lstinputlisting command is used to insert an external file containing
% code into the document formatted in the same manner as a figure or table.
% All stand alone listings should have a label and caption. You can optionally
% specify a programming language to enable syntax highlighting.
\lstinputlisting[label={lst:my_label_name}, caption={The caption.}]{the_file}
\lstinputlisting[language={the_language}, label={lst:the_label}, caption={The
    caption.}]{the_file}

% An example showing how Listing 3.1 is formatted in LaTeX code.
% The C code is stored in its own file as C code, allowing it to be modified
% and prepared separately using a dedicated code IDE to ensure correctness and
% proper formatting.
\lstinputlisting[language={c}, label={lst:c_hello_world}, caption={An
    implementation of an important algorithm from our work.}]{./listings/
    hello_world.c}
```

Listing 4.3: Examples of methods for typesetting code listings within a LaTeX document.

4.6 Tables

Tables are also quite predictably captioned and formatted the same way. It is important to decide on a style for how you will organize your data and apply that style consistently for all of your tables. Table 4.1 shows one possible way of styling your data but is by no means the only way of doing so neatly. Consistency is the key.

Table 4.1: An example of a table formatted with caption.

Some	Relevant	Fields	From	Your	Data
0	0	0	0	0	0
1	1	1	1	1	1
2	2	2	2	2	2

Chapter 5

Conclusions and Future Work

In this document we have demonstrated the use of a LaTeX thesis template which can produce a professional looking academic document.

5.1 Contributions

The main contributions of this work can be summarized as follows:

- **A LaTeX thesis template**

  Modify this document by adding additional top level content chapters. These descriptions should take a more retrospective tone as you include summary of performance or viability.

- **A typesetting guide of useful primitive elements**

  Use the building blocks within this template to typeset each part of your document. Aim to use simple and reusable elements to keep your document neat and consistently styled throughout.

- **A review of how to find and cite external resources**

  We review techniques and resources for finding and properly citing resources from the prior academic literature and from online resources.

5.2 Future Work

Future editions of this template may include additional references to Futurama.

Bibliography

[1] Things Tech, “Which programming language to start with as a beginner,” 2020, [Online; accessed August 10th, 2020]. [Online]. Available: https://thingsteck.wordpress.com/

[2] Internet Live Stats. (2020). [Online]. Available: https://www.internetlivestats.com

[3] G. Branwen. (2020) Internet search tips. [Online]. Available: https://www.gwern.net/Search

[4] RELX Group. (2019) Mendeley. [Online]. Available: https://www.mendeley.com

[5] J. T. Kajiya, “The rendering equation,” in *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH ’86. New York, NY, USA: ACM, 1986, pp. 143–150.

[6] Apple Inc. (2020) Swift. [Online]. Available: https://developer.apple.com/documentation/swift

[7] M. Potuck. (2020) Apple hits 1.5 billion active devices with 80iphones and ipads running ios 13. [Online]. Available: https://9to5mac.com/2020/01/28/apple-hits-1-5-billion-active-devices-with-80-of-recent-iphones-and-ipads-running-ios-13/

[8] K. Finley. (2020) Python is more popular than ever. [Online]. Available: https://www.wired.com/story/python-language-more-popular-than-ever/

[9] B. Popper. (2020) The 2020 developer survey results are here! [Online]. Available: https://stackoverflow.blog/2020/05/27/2020-stack-overflow-developer-survey-results/

[10] A. Goel. (2020) Best programming language to learn in 2020 (for job and future). [Online]. Available: https://hackr.io/blog/best-programming-languages-to-learn-2020-jobs-future

[11] Stack Overflow. (2013) Python vs cpythony. [Online]. Available: https://stackoverflow.com/questions/17130975/python-vs-cpython

[12] Python Software Foundation. (2020) What is python? executive summary. [Online]. Available: https://www.python.org/doc/essays/blurb/

[13] ——. (2020) Comparing python to other languages. [Online]. Available: https://www.python.org/doc/essays/comparisons/

[14] Overleaf. (2020) Overleaf documentation. [Online]. Available: https://www.overleaf.com/learn

[15] Stack Overflow. (2008) Tex stackexchange. [Online]. Available: https://tex.stackexchange.com

[16] A. Ravikumar. (2019) Texnique. [Online]. Available: https://texnique.xyz

[17] J. Whittle. (2015) Path traced glass dragons.

Appendix A

Implementation of a Relevant Algorithm

```
#include <stdio.h>

int main(int argc, char *argv[]) {
  printf("Hello world.\n");
  return 0;
}
```

Listing A.1: An implementation of an important algorithm from our work.

Appendix B

Supplementary Data

The results of large ablative studies can often take up a lot of space, even with neat visualization and formatting. Consider putting full results in an appendix chapter and showing excerpts of interesting results in your chapters with detailed analysis. You can use labels and references to refer the reader here for the full data.